Practical Machine Learning

Project Wrapup by Rebecca Lu

How I built the model

The goal is to predict the manner in which users of FuelBand and Fitbit and etc did the exercise. Based on their behaviors, we are going to group them into 5 categories, A, B, C, D, and E. I want to build a prediction model that could maximize the accuracy and minimize the out-of-sample error. And I am going to preprocess the data, eliminating the attributes which have NA values and missing data. After preprocessing, I will divide the data set into 2 subsets: a training set and a testing set. Then I will use the Random Forest algorithm to perform the pattern classification. After that, accuracy of the model and the expected out-of-sample error will be calculated.

Seed is set to be 7, so that the result could be reproduced.

Corss Validation

I will subsampling the training data into a subtraining data (75% of the training data set) and a subtesting data. I will build the model based on the subtraining data set and see how it works on the subtesting data set. This way we can avoid the problem of model overfitting.

Expected out-of-sample error

A prediction model which is not over-fitted is always expected to have some out-of-sample error.
The expected out-of-sample error corresponds to the misclassified observations/ total number of observations. Therefore it is related to the quality of my model and the total numbers of observations.

Why I made the choices I did

Random forest are an ensemble learning method broadly used for classification and regression. It is operated by constructing a multitude of decision trees and then make predictions based on the mode of the trees. Tree emsembles, in particular Random Forests, are easy to tune. It is fast and scalable. Therefore building a Random Forest model will be a good choice for this pattern classification case.

R Codes and Analysis

Loading Libraries and Reading Data:

```
library(caret)
library(ggplot2)
library(randomForest)
getwd()
```

```
## [1] "C:/Users/Rebecca/Dropbox/Personal + ebook/Practical Machine Learning Coursera"
```

```
setwd("C:/Users/Rebecca/Dropbox/Personal + ebook/Practical Machine Learning Coursera/Project")
set.seed(7)
```

Then read the data from the local file. We're gonna mark empty string, NA, and Null as NA to make sure the training data set and testing data set have the same levels.

```
testing <- read.csv('pml-testing.csv', na.strings=c("", "NA", "NULL"))
training <- read.csv('pml-training.csv', na.strings=c("","NA","NULL"))
dim(testing)
```

```
## [1]  20 160
```

```
dim(training)
```

```
## [1] 19622   160
```

Preprocessing:

Discard attributes which are all NAs and eliminating the unrelated attributes such as 'user_name', 'raw_timestamp_part_1' and etc.

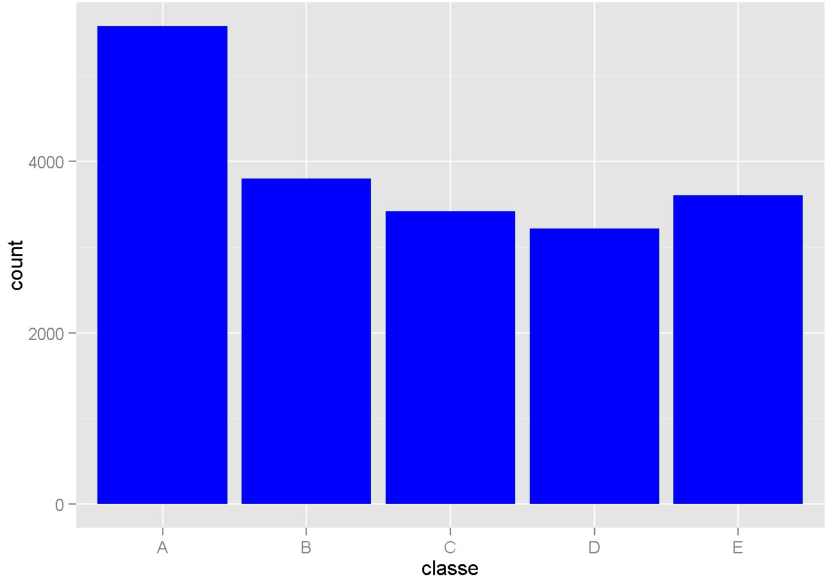
```
testing1 = testing[, colSums(is.na(testing)) == 0]
training1 = training[, colSums(is.na(training)) == 0]
testing2 = testing1[, -c(1:8)]
training2 = training1[, -c(1:8)]
ncol(testing2)
```

```
## [1] 52
```

```
ncol(training2)
```

```
## [1] 52
```

```
# to get a general idea of what the distribution of training classe looks like, I am going to use ggplot2 packa
ge to draw the histogram:
ggplot(training2, aes(classe)) + geom_histogram(fill='blue')
```



SubSampling:

Divide the training data into two parts: a subTraining data set and a subTest data set.

```
partition <- createDataPartition(training2$classe, p = 0.75, list=FALSE)
subTraining <- training2[partition ,]
subTest <- training2[-partition ,]
```

Build Random Forest Model and Evaluate Out-of-Sample Accuracy:

Build the Random Forest Model using the subTraining data.

```
rfModel <- randomForest(classe ~ ., data=subTraining, ntree=10)
```

Predict the subTesting data and see how it works.

```
predictSubTest <- predict(rfModel, subTest)
confusionMatrix(predictSubTest, subTest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1391    9    1    0    0
##          B    2  934    6    0    0
##          C    1    6  846   14    1
##          D    1    0    2  787    3
##          E    0    0    0    3  897
##
## Overall Statistics
##
##                Accuracy : 0.99
##                  95% CI : (0.9868, 0.9926)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9874
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9971   0.9842   0.9895   0.9789   0.9956
## Specificity            0.9972   0.9980   0.9946   0.9985   0.9993
## Pos Pred Value         0.9929   0.9915   0.9747   0.9924   0.9967
## Neg Pred Value         0.9989   0.9962   0.9978   0.9959   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2836   0.1905   0.1725   0.1605   0.1829
## Detection Prevalence   0.2857   0.1921   0.1770   0.1617   0.1835
## Balanced Accuracy      0.9971   0.9911   0.9920   0.9887   0.9974
```

We can see from the above result that we got an Accuracy of 0.99 and the 95% CI is (0.9868, 0.9926). Therefore our model is pretty good in terms of predicting the patterns.

Predict on Testing Data:

```
predictTesting <- predict(rfModel, testing2)
predictTesting
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  A  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

Save the prediction to files for submission:

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
pml_write_files(predictTesting)
```

Reference

https://en.wikipedia.org/wiki/Random_forest (https://en.wikipedia.org/wiki/Random_forest)